Task UR40.2.1
CDRL #01170 - "IMPLEMENTATION OF RLF ENHANCEMENTS"
Version X1-0

Version Description Document

1. SCOPE

This document provides an overview for the R-increment enhancements to the Reusability Library Framework (RLF) software. This software was developed in its original form under the STARS Foundations program (contract number N00014-88-C-2052, administered by the Naval Research Laboratory). This delivery includes modifications to the RLF software in several key areas which are briefly described below. Additional information can be found in README files within the various subdirectories that comprise this delivery. In particular, these README files contain information necessary to install the software for evaluation purposes. The user manuals for each of the major subsystems included in this delivery remain in their original STARS Foundations form. These manuals, though out-of-date, are included to help the inexperienced user become familiar with the functionality provided within the RLF. Updated versions of these manuals are scheduled for delivery in early May 1990.

1.1 Identification

Reusability Library Framework (RLF).

Chief Programmer for the work performed under the current contract is James Solderitsch. He can be reached at:
Unisys, Valley Forge Operations, PO BOX 517, Paoli PA 19301-0517.
Telephone: 215 648 2831. Electronic mail: jjs@prc.unisys.com

1.2 System Overview

The Reusability Library Framework (RLF) is a system designed and implemented to support the production and installation of domain-specific software library systems. The RLF is based on two fundamental subsystems: AdaKNET (Ada Knowledge NETwork) and AdaTAU (TAU is an acronym for Think Ask Update) which are knowledge representation and inferencing systems derived from systems previously developed at Unisys in Prolog. These subsystems are supported by an integrating framework to allow them to be used in combination with each other. AdaKNET and AdaTAU are also equipped with interface specification languages (SNDL and RBDL respectively) that are used to generate Ada code to initialize domain models that describe the library (or application) domain. In addition to the support of library systems, the RLF was used to develop a proof-of-concept Ada unit test assistant during the STARS Foundations period and is currently being considered for the representation of software and reuse process models which are themselves machine processable.

The top level directory of this delivery is partitioned into nine (9) subdirectories, each of which is described briefly below. More detailed instructions on building the RLF components can be found in each of the subdirectories.

The nine subdirectories are: Common, Adatau, Adaknet, Rbdl, Sndl, Sndl_specs, Rdbl_specs, Hybrid and Librarian.

Common contains low-level data abstractions and utilities shared by the other RLF subsystems. For example, basic set and list abstract data types (ADT's) are located in the Common directory. Some implementation

dependent parts of Common need modification in order to build the system (see the AdaTAU and AdaKNET README's).

Adatau contains the source code for the AdaTAU subsystem. AdaTAU is a rule-based inferencer, and may be used as a stand-alone knowledge representation system, or in conjunction with AdaKNET (as in the Librarian application).

Adaknet contains the source code for the AdaKNET subsystem. AdaKNET is a structured-inheritance knowledge representation system. Like AdaTAU, it may be used as a stand-alone system. The AdaKNET source also includes an interactive AdaKNET browser-editor application.

The Rbdl and Sndl directories contain source for the Rule Base Definition Language (RBDL) and Semantic Network Definition Language (SNDL) translators. These translators transform high-level, non-procedural descriptions of rule-based and semantic network knowledge bases into Ada programs which call AdaTAU and AdaKNET interfaces to initialize knowledge bases. Most of the code in these directories was automatically generated using a language processor generation system. This generator is not included in this delivery.

Sndl_specs contains the SNDL specifications for the knowledge bases used by the Librarian application. These specification files must be processed by the SNDL translator to produce programs to build the networks used by the Librarian. This process of initializing the knowledge bases provides an excellent means of testing the installation of AdaKNET, and SNDL.

Rdbl_specs contains the RBDL specifications for the knowledge bases used by the Librarian application. These specification files must be processed by the RBDL translator to produce programs to build the rule bases used by the Librarian. This process of initializing the knowledge bases provides an excellent means of testing the installation of AdaTAU, and RBDL.

Hybrid contains the source code for the hybrid knowledge representation system used by the Librarian application. This hybrid system combines AdaKNET and AdaTAU into a structured-inheritance semantic network with rule-base inferencing capabilities.

Librarian contains the source code for the Librarian and its other functional roles.

2. RELATED SOFTWARE

Later work under UR40 will address the integration of the RLF with the baseline SEE which itself is based on CAIS-A. The RLF is also a potential customer for the generic graphical browser which is slated for development during the latter stages of UR20.

Because the RLF is targeted to the support of software libraries, and by extension software repositories, the software repository systems under development by the other STARS contractors are related to the RLF.

Software development tasks addressing the representation and establishment of process models are also related to the software being developed during UR40.

3. VERSION DESCRIPTION

This is the first version of the RLF tc be delivered that was supported by the STARS Prime program. As such this version should be considered version 1.0, with any future deliveries identified by appropriate decimal version numbers greater than 1.0. It is anticipated that during the R increment, future deliveries will be relatively minor extensions of version 1.0 so that no increase in the base version number are planned.

3.1 Inventory of Contents

To minimize the occurrence of redundant information, a complete file inventory is not included in this document. The various README files that appear in the subdirectories of this delivery contain Ada compilation order listings which should agree with the contents of each of the subdirectories. The collected files in each of these subdirectories are the complete inventory.

3.2 Changes Installed

Version 1.0 is an initial delivery of the RLF under the STARS Prime program. However, the following paragraphs outline the changes that have taken place since the RLF was delivered to the NRL at the close of the STARS Foundations project (March 1989).

Modifications to AdaTAU rule-based inferencing subsystem:

1) Modification of AdaTAU constructs, RBDL syntax, and the RBDL translator's code-generating backend to provide for declaration of a new type of fact parameter, called a 'focal' fact, to be passed between separate inferencers when an Frule (Focus rule) directs a change of inferencing context while using Distributed AdaTAU (DTAU). Includes modification of the syntax of a RBDL Frule to include a list of facts to be exported when that Frule's suggestion to switch context is taken. This also required restructuring of the main procedure of DTAU to handle the improved context-switching mechanism.

   This change in AdaTAU permits more concise descriptions of the effect of switching an inference context from one inference base to another, By including a list of facts to be asserted to the current inferencer's fact base, and by passing some of these facts to the destination inferencer's fact base through corresponding fact parameters, the knowledge base designer can precisely control the flow of information among inferencers. This change also increases the efficiency of a context switch between inferencers.

2) The code-generating backend of the RBDL translator was modified so that it would generate a complete Ada subprogram which builds an AdaTAU inference base and then makes it persistent in file form. This greatly reduced the complexity of building and installing AdaTAU inferencers.

3) The basic inference cycle was modified to allow user-interruptible inferencing, and an explanation mechanism was added to supply the user with the justifications for inferencing. The justifications are defined in an inferencer's RBDL specification. The change causes the justification clauses included within a RBDL specification to be viewable as inferencing proceeds. The AdaTAU user is also able to control the level of explanations to be provided to the application user.

4) New specifications adding non-monotonic reasoning to AdaTAU. These changes will allow AdaTAU to assert facts as well as retract facts as a consequence of an answer to a question, or

as a consequent of an inference rule meeting its antecedents. RBDL was modified to support the statement of rules that express fact retractions as well as assertions. Non-monotonic inferencing provides a much more realistic view of the way a human will interact with a computer-based inference system. The facts stored within the system will be able to be more accurately adjusted as the application end-user interacts with the system.

5) Modifications to existing persistence operations to handle non-monotonic information.

6) Modifications for asserting and deleting facts from a given fact base.

7) Some Ada packages were reorganized to obtain improved compiler performance and to work around compiler limitations.

Modifications to AdaKNET structured inheritance network subsystem:

1) Operations were added to remove role restrictions and to query the existence of role restrictions. Previous implementations of role restrictions were too limited in the way these restrictions were processed. The AdaKNET user interacting with the network is now able to adjust role restrictions more freely. Role restrictions are further explained in the AdaKNET user's manual.

2) Role differentiation was implemented. Roles are in essence the attributes of objects (concepts in AdaKNET terminology) that collectively categorize the object. In modeling complex relationships, it is useful to be able to more finely categorize potential roles. For example, if a software component is known to require a software context consisting of related software components, it is useful to be able to categorize some of these components as belonging to one group (e.g. machine dependencies) while others belong to a different group (e.g. compiler dependencies). All of these components all have the same role, namely that of being *required* subcomponents.

   Role differentiation enables the modeler to introduce two or more different subroles each of which represents a category of the parent role's potential role instantiations. The instantiations for these subroles must, however, remain semantically consistent with the parent role.

3) Some minor cleanup and performance enhancement was done. For example, the body of the Adanets package was broken up into separate compilation units.

4) SNDL was changed to allow independent restriction of roleset types and ranges (to reflect the roleset changes mentioned under 1 above).

5) SNDL's topological sort was changed so that it would generate more efficient code.

Modifications to the hybrid knowledge representation:

1) Addition of support for integer and text objects' association with concepts of the AdaKNET semantic network. Previously, only AdaTAU inferencers could be associated with network concepts. Includes re-organization of the hybrid state layer and addition of packages containing operations on all three types of state. By allowing attached integer and textual values to concepts in a semantic network, the modeler is able to keep the network structure from becoming cluttered with information that has no bearing on the

information handled via the structured inheritance network. The textual information can be maintained as simple text files which are built and modified with an ordinary text editor.

Modifications to current RLF terminal-style interface:

1) Modifications to the presentation of information from the Library_Browser application including re-wording of command requests and menu selections to improve the 'conversational style' of the interface, restructuring of information presentation to fit within screen space limits, and automatic pausing of screen scrolling. These changes allow information recorded within an RLF-based system to be effectively displayed on devices with limited display capabilities.

2) Addition of interrupt handling to catch and manage control-key sequences to prevent corruption of the knowledge base following an improper exit and to allow discontinuation of unwanted menus.

3) Addition of menu-driven traversal of the AdaKNET semantic network and its attributes to improve the user's mobility in the library.

Modifications to Library_Browser application structure:

1) Division of the Librarian application into two different Library_ Browsers, one for the library user (User_Browser) and one for the library administrator (Admin_Browser). The Admin_Browser contains all browsing and editing capability, while the User_Browser only allows browsing and not modification of the reuse library. This separation will enable the safeguarding of sensitive information and prevent casual users from modifying the knowledge bases.

2) Simplification of the strategy used for Distributed AdaTAU intelligent guidance in the Library_Browser permitted removal of the Advising_Inferencer package of the Librarian application. This change reduced the number of layers needed to implement the advising mode of the librarian application thereby saving both some memory and execution time.

3) Incorporation of modified operations at the AdaTAU and AdaKNET levels into librarian application code.

3.3 Adaptation Data

The README files warn of several installation site-specific dependencies. These READMEs should be consulted before attempting to install the software. All past development activity has taken place in a UNIX environment. An early STARS Foundations version of the software, did compile and execute on Unisys PC compatible with extended memory and a large hard disk. However, the RLF team has not attempted to build or use the system in its current form outside of a UNIX environment. Any experiences of users of this software in non-UNIX environments should be forwarded to the chief programmer.

3.4 Interface Compatibility

Does not apply to initial delivery.

3.5 Summary of Change

The changes outlined in 3.2 affect the expressibility available in both

the AdaKNET and AdaTAU subsystems as well as the operational performance of the librarian and related applications. These changes will be documented in the updated user manuals to be delivered later this year. The current delivery is version 1.0 from which all future versions of the RLF will be derived.

3.6 Installation Instructions

Installation of the RLF software is covered in the README files local to each of the subdirectories contained in this delivery.

3.7 Possible Problems

Additional testing of this software will occur over the next several months. At the time of the delivery of version 1.0, successful compilation and execution of the software was obtained on Sun Workstations using Sun OS 3.5 and the Verdix Ada Development System. In conjunction with the final demonstration, execution under Sun OS 4.0 and at least one other Ada compilation system will be verified.

This task does not include the development of enhancements to the librarian domain model which is used to demonstrate and test the feature set of the basic knowledge representation systems. It is possible that as more sophisticated models are developed, limitations and other problem areas will be exposed. These will be addressed as long as resources are available for this purpose.

3.8 Enhancements

In addition to the enhancements that are outlined above, several other enhancements have been identified as providing significant value in the production and maintenance of domain models. In particular, two which have been considered for near-term implementation are Application Rules and an AdaTAU browser/editor capability. Application Rules enable the statement of rules that can interact with the operations available within the application that is using AdaTAU, and thereby update the state of the knowledge base for the application based on direct input from the application (in much the same way as Qrules in the current RLF enable user interaction to lead to knowledge base modifications). An AdaTAU browser editor will permit dynamic fine-tuning of inferencer specifications and permit small-scale modifications to be made without regenerating an inference base from scratch.

Recently memory utilization and user interface issues have surfaced in conjunction with an internal Unisys user of the STARS Foundations RLF software. These issues have suggested additional enhancements to improve both the use of dynamic (pointer-referenced) memory as well as more sophisticated user interfaces both for remote, terminal-based users as well as users of workstations capable of graphical displays.

4. NOTES

The reader should note that both AdaTAU and AdaKNET were designed for independent use by applications requiring knowledge representation and inferencing capabilities. The specification languages provided for these subsystems foster their transfer to diverse application areas and their programmatic interfaces enable their integration into general Ada applications. Future work will address performance issues such as memory requirements and execution speed. Additional using applications will help determine system shortcomings and lead to their correction.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	27 July 1990	Version Description Document

4. TITLE AND SUBTITLE
Reusability Library Framework (RLF)

5. FUNDING NUMBERS
STARS Contract
F19628-88-D-0031

6. AUTHOR(S) James J. Solderitsch, William Bredbenner, Timothy Schreyer, Ray McDowell, Keith Cassell, Kurt Wallnau

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Unisys Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

8. PERFORMING ORGANIZATION REPORT NUMBER
GR-7670-1165(NP)

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Department of the Air Force
Headquarters, Electronic Systems Division (AFSC)
Hanscom AFB, MA 01731-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
01170

11. SUPPLEMENTARY NOTES
There are three related RLF reports: (RLF) AdaKNET User's Manual, (RLF) AdaTAU User's Manual and (RLF) Librarian Software User's Manual

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release;
distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Reusability Library Framework (RLF) is a software system designed and implemented to enable the development of application-area (or domain) specific reuse libraries. The framework has been implemented in Ada, using basic principles of software engineering, and is both extensible and integrable with other Ada systems. The RLF includes a semantic network subsystem and a rule-driven inferencing subsystem which are used to record domain knowledge and heuristics in a form suitable for use within Ada applications. Besides the creation of several kinds of librarian applications, the RLF has been used to represent process model descriptions, capture Ada and heuristic testing information for use by an Ada unit test plan generator, and to actively aid a documentation specialist in producing structured technical documentation from templates and a knowledge-based model of the document production software being used.

14. SUBJECT TERMS
Reusability Library Framework (RLF)
AdaKNET AdaTAU Librarian

15. NUMBER OF PAGES
123

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR